Experience based localization in wide open indoor environments

Rahul Gautam*, Harsh Jain, Mayank Poply, Rajkumar Jain, Mukul Anand, and Rahul Kala

Abstract: This paper solves the problem of localization for indoor environments using visual place recognition, visual odometry and experience based localization using a camera. Our main motivation is just like a human is able to recall from its past experience, a robot should be able to use its recorded visual memory in order to determine its location. Currently experience based localization has been used in constrained environments like outdoor roads, where the robot is constrained to the same set of locations during every visit. This paper adapts the same technology to wide open maps like halls wherein the robot is not constrained to specific locations. When a robot is turned on in a room, it first uses visual place recognition using histogram of oriented gradients and support vector machine in order to predict which room it is in. It then scans its surroundings and uses a nearest neighbor search of the robot’s experience coupled with visual odometry for localization. We present the results of our approach test on a dynamic environment comprising of three rooms. The dataset consists of approximately 5000 monocular and 5000 depth images.

Keywords: localization, visual place recognition, experience based localization, machine learning

1 Introduction

With the growing industry of self-driving cars, humanoid robots, and industrial robots there is a need for robots to become autonomous and be able to navigate themselves in any environment. For this purpose, we need precise and accurate robot localization and tracking algorithms. Many of these algorithms are inspired by the human visual system and use visual hardware such as stereo systems or RGBD (Red-Green-Blue-Depth) sensors.

Localization is the basis of every robotic application. In order for any robot to do its required task, it should answer the question “where am I?” All localization techniques determine the robot’s location and orientation in the given environment. The process of mapping cannot be separated from the process of localization, as localization is usually performed with respect to the surroundings or the environment. Maps are usually represented in two frameworks: the metric framework which is a geometric model of the map and is the most commonly used framework; and a topological framework which considers the places and the relations existing between the places.

Experience based localization is a recent technique used to solve the problem of localization, wherein the past experiences of the robot are stored and used for answering localization queries. The technique is widely used in a topological or constrained environment wherein the robot travels to nearly the same position every time. The most common example is self-driving cars and corridors wherein there is a very little lateral dispersion that the robot may show on the road or corridor, and therefore at every instant of travel the percept is the same. This paper solves the problem of localization for wide open indoor environments wherein the robot could be anywhere inside a big hall. We also present a hybrid approach of localization using both experienced based localization and visual odometry.

The main aim of this paper is that whenever a robot is switched on at an unknown location, the robot should be able to identify its location. This problem is decomposed into finding out which room it is in, and its location and orientation in that room. This paper aims at solving the problem of indoor localization on a previously built static map represented using a topological framework. We use past information/experience from the robot in order to recognize the room in which the robot is present using visual place recognition. After that we perform experience based localization by extracting out only the experiences that are relevant from the database using a nearest neighbor classifier and perform visual odometry on the images.
We use visual odometry so that we can adjust to some dynamic changes in the environment for which our experience based localization algorithm might not provide the accuracy required. The entire process is summarized in Figure 1.

Figure 1: Project Overview: This figure shows the entire process of the project approach starting with visual place recognition, followed by finding the starting point and experience based localization operating in parallel with visual odometry.

2 Literature survey

Lowe [1] extracted SIFT features for visual place recognition. The authors matched SIFT features against a database of features using a Hough transform and a fast nearest neighbor algorithm. Finally the best pose parameters verification was done through least-squares. Lowry et al. [2] created a topological graph and then performed localization via SIFT feature matching and for next frames search is only done among the neighbors of the detected node. The paper also presented the process of navigation in the topological graph.

Linegar et al. [3] used both visual odometry and visual place recognition in order to localize a robot in a time and weather varying environment. If visual place recognition failed, the robot was localized using visual odometry, till the time it was able to re-localize itself in the environment and continue with localization using visual place recognition. The process led to the creation of sophisticated and robust maps. The authors presented localization results of a car driven on a 37 km track.

Napier et al. [4] demonstrated online vehicle pose estimation with only the use of a stereo camera. The approach used visual appearances of the scenes in order to perform pose estimation of the vehicle along with the visual SLAM generated trajectory. Fiala and Ufkes [5] performed visual odometry by matching 3-D points using corresponding 2-D descriptors for images. The paper used 3-D data along with SIFT or SURF features to find the camera’s orientation and translation by using RANSAC (random sample consensus) algorithm thus performing live visual odometry.

Pronobis et al. [6] used an appearance based approach for localization working under different illuminations and over a large span of time. The approach used a rich descriptor composed of a high dimension histogram of the image and a support vector machine (SVM). Shi and Thomasi [7] discussed the problem of selecting feature points corresponding to physical world points which are good to track. The authors explained how usually points with big eigenvalues are sufficient to be detected as corners in images, however they may not necessarily be corners in the physical world. The authors modified the scoring function from Harris corner detector [8] to obtain better features for tracking.

Luo et al. [9] demonstrated an adaptive visual place recognition algorithm which was able to learn from its experience and continuously adapt to changes in the environment. The recognition algorithm was able to adapt to changes in its database and update it. It used incremental SVM in order to keep the algorithm running within the memory requirements. Pronobis et al. [10] showcased the ability of software to learn in realistic settings. The authors used incremental SVM based learning in order to build recognition models of the environment. Pronobis and Caputo [11] approached the visual place recognition problem along with a degree of confidence in the answer. The authors used a SVM classifier in order to classify the hypothesis and distance from the SVM classification hyperplane in order to get the confidence of the hypothesis.

Some other related approaches are also briefly discussed. Llorca et al. [12] presented a vehicle logo recognition algorithm using a sliding window combined with a histogram of oriented gradients (HOG) and SVM classifier. Shimosaka et al. [13] presented an indoor localization mechanism with the use of Zig-Bee devices. Kala [14] described the different localization schemes in a self-driving car technology and explained the importance of localization in planning and decision making. Kadota et al. [15] explained the problem of pedestrian recognition on an embedded system using a simplified HOG algorithm. Henry et al. [16] built dense 3-D maps of indoor environments. The paper used visual and depth information combined with loop closure detection and finally optimized the poses obtained in order to build maps which are globally consis-
tent. Dryanovski et al. [17] used an RGBD sensor and then used real-time visual odometry and mapping. The authors used visual odometry with a Kalman filter. The authors used a registration algorithm which was able to detect small loops online. Irie et al. [18] presented an outdoor stereo vision based localization system. The system took care of illumination changes by forming 2-D occupancy grid maps from the 3-D point clouds. The robot’s pose was estimated using a particle filter combined with visual odometry and map matching.

Kaundal et al. [19] localized a fixed node by pursuit nodes in outdoor localization using two different methods: first using LNSM (Log Normal Shadowing Model) which used the ZigBee protocol and the received signal strength for localization; and the second using a method based on Hybrid TLBO (Teacher Learning Based Optimization Algorithm)- Unilateral technique. The paper also compared the localization results for both the methods and presented them. It also showed that the fixed node becomes 100 per cent discoverable by use of the second method. Rathnam and Birk [20] presented a 3-D exploration algorithm using a group of robots which are always within communication range of each other. Since the method is based on a greedy computation strategy using a heuristic function, it is computationally efficient. The paper also presented an efficient strategy to recover from deadlocks resulting from local minimums. The exploration algorithm was tested on a simulator for autonomous underwater vehicles (AUV) taking into account their sensors. The effect of increasing and decreasing the number of robots and the communication range of the robots on the algorithm was also investigated.

Li et al. [21] presented an adaptive clustering technique based on exploration and exploitation strategies in contextual multi-armed bandit settings. This was to improve the performance of context based filtering and classic collaborative filtering methods which do not perform well under dynamic recommendation domains such as news recommendation and computational advertisement. The algorithm used the collaborative effects arising between the interactions of users with the items and then dynamically grouped the users based on the items under consideration. Li [22] also proposed algorithmic solutions to the networked bandit problem so as to integrate the strong social components in the bandit algorithms which increased the performance of the algorithm drastically. Starting from the global Laplacian strategy where each node has its own bandit algorithm and is allowed to share signals with its neighboring nodes, the paper aimed to develop and experimentally tested more strategies of clustering nodes which were more scalable. Li and Kar [23] proposed a context-aware clustering of bandits (CAB) algorithm which captured collaborative effects. CAB dynamically clustered the users based on the content universe in situations such as the real-world recommendation system where multi-armed bandits performed pretty well. Korda et al. [24] provided algorithms for solving the linear bandit problem in peer to peer networks with limited communication capabilities. First the authors assumed all peers solved the same linear bandit problem and prove that the algorithm achieves optimal asymptotic regret rate. Then the authors assumed that within clusters there were clusters of peers solving the same bandit problem and the algorithm discovered these clusters while within each cluster it achieved the optimal asymptotic regret rate.

Kar et al. [25] presented online stochastic algorithms for quantification-specific measures in order to perform class prevalence which is what fraction of the population belongs to a particular class. The paper also presented hybrid algorithms to balance out quantification and classification performance. Yoo et al. [26] proposed a semi-supervised machine learning algorithm. It improved localization performance as training data comes from received signal strengths of wireless communication link and thus reduced the needs of calibrating labelled data from the unlabeled data. The algorithm was used for evaluating the position of a smartphone-based mobile robot. The experimental results showed that the algorithm did not compromise the computational speed and was more robust compared to state-of-the-art semi-supervised learning algorithms. Park and Roh [27] presented a global localization technique based on a 2-D range scan place recognition technique. The paper used a SVM to train a set of classifiers for place recognition. The paper used a bag-of-words approach to create the feature vectors for training the SVM classifiers for place recognition. The algorithm first produced coarse localization for selecting the best candidate places where the robot may be located based on place recognition. After that fine localization was done via fast spectral scan matching and a particle filter algorithm. The paper also presented the results of extensive simulations and experiments.

Lu et al. [28] presented an image-based indoor localization system based on thermal imaging which can be used in case of emergencies such as a blackout. Learning was applied in order to enrich the thermal imaging classification. Active learning enhanced the performance of the algorithm and the algorithm was able to properly localize the robot in a dark environment. Winterhalter et al. [29] presented an accurate indoor localization method for RGBD smartphones or tablets. The method uses the 2-D outline of the environment from the architectural draw-
ings as the map in order to perform localization. It used data from the sensors present on the device to handle differences between the map and real world. The algorithm used a particle filter to estimate the 6 DoF pose of the device using the sensors. Results of the localization approach are also presented on a Google Tango device.

Most of the abovementioned techniques have been tried in outdoor scenarios wherein the map is topological in nature and the vehicle can show a very small lateral movement. This means the experience will be the same every time the vehicle passes the same regions. The algorithms are adaptive to corridor like situations where the map also is topological. For halls and other rooms with wide open spaces, the robot may be found at any place and the same place may be different every time the robot passes by. This creates a very large number of possible experiences. Such a setting is not actively researched. Furthermore, the proposed approach integrates visual place recognition with experience based localization to limit the number of experiences to be considered during search. In other cases wherein wireless sensors are used, the wireless sensors are known to be prone to noise. Other algorithms which propose effective localization do not give localization approaches when the robot is travelling short distances. In this case visual odometry becomes far more efficient than experience based localization.

3 Background

3.1 Good features to track

Good features to track (GFTT) is similar to harris corner detection [30] where we move a window over the entire image and find values where Eigen vectors have maximum values which are found by finding the derivatives along the X and Y directions. Shi and Thomasi made modifications to the original formula such that tracking is improved and can overcome problems such as occlusions and features that do not correspond in the real world. The proposed work uses GFTT features for selecting the features in the images to compute visual odometry between two images.

3.2 Visual odometry

Using visual odometry we estimate the pose of an object which in our case is represented by the X-coordinate, the Y-coordinate and the orientation $\theta$ which is measured in the anti-clockwise direction from the X-axis. The PnP (Perspective-n-Point) [31] problem is estimating the pose of a calibrated camera from given sets of 3-D points. We follow the below steps in order to perform visual odometry between two images:

1. Take input images from the camera or any other input stream. Extract GFTT features and descriptors from the image. First we select the features in the images. The GFTT features selected in the image for a sample image are shown in Figure 2.

![Figure 2: GFTT Features. The green dots show the GFTT feature points detected in our image.](image)

2. Match the descriptors between the images using the FLANN (Fast Approximate Nearest Neighbor Search Library) matcher. Now, we see the correspondences between the feature points in our image and the world coordinates as shown in Figure 3. The figure also shows how 3-D points are mapped to corresponding (u, v) pairs between the images via red lines, i.e., the 2-D projections of these 3-D points in the camera coordinate system.

3. Perform pose estimation between the images using PnP and RANSAC [32]. We aim to retrieve the pose (rotation $R$ and translation $T$) using these parameters and the focal length of the camera. During the whole process of computing visual odometry we use the pinhole model of the camera as shown in Figure 4.
Rahul Gautam et al.

Figure 3: Transformation from 2-D image to 3-D world coordinates. This image shows the correspondence between points on the image \((u, v)\) and in the physical world, i.e., \(P_i\).

Figure 4: Pinhole Camera Model.

We first project the obtained features into their corresponding 3-D points using (1) and (2):

\[
sm^T = A [RT] M^T \tag{1}
\]

\[
\begin{bmatrix}
 s \\
 v \\
 1 
\end{bmatrix} =
\begin{bmatrix}
 f_x & 0 & c_x \\
 0 & f_y & c_y \\
 0 & 0 & 1 
\end{bmatrix}
\begin{bmatrix}
 r_{11} & r_{12} & r_{13} & t_x \\
 r_{21} & r_{22} & r_{23} & t_y \\
 r_{31} & r_{32} & r_{33} & t_z 
\end{bmatrix}
\begin{bmatrix}
 X \\
 Y \\
 Z \\
 1 
\end{bmatrix} \tag{2}
\]

Where \(s\) is the scaling factor, \((X, Y, Z)\) are the world coordinates projection of the point pixel \((u, v)\) in the image obtained via a camera having \((f_x, f_y)\) as the focal length and \((c_x, c_y)\) as the principal center. \(A\) represents the camera matrix, i.e., the matrix of the camera’s intrinsic parameters. The matrix \([R|T]\) is called as the joint rotation and translation matrix. It is used to describe the camera motion in a static scene.

After we have obtained the 3-D points corresponding to the 2-D features we find the motion by applying PnP RANSAC. RANSAC is a process through which we are able to take care of these outliers by iteratively selecting a subset of points non-deterministically and finding inliers (points which have correct 3-D corresponding points) in them. We continue this same process many times until an approximate good data of inliers can be obtained. We usually select sufficiently high number of RANSAC iterations so that probability of finding inliers becomes high. After we are done with finding the set of inliers we proceed to the final step i.e., the camera pose estimation via PnP algorithm. The PnP algorithm returns the camera pose by finding the differences between inliers of two successive images and outputs the final rotation and translation vectors that the camera has gone through. The process is explained in Figure 5 and Algorithm 1.

**Algorithm 1: Visual odometry**

1. Initialize_Camera using Calibration_File
2. \(\tau(0) \leftarrow [R_z(\theta_0) (x_0, y_0) 0]^T : 0 0 0 1]\) # \(\theta\) is initial angle
3. \(t=0, I(t) \leftarrow 3Dsensor_Capture()\)
4. while (3Dsensor is ‘ON’): 
5. \(I(t) \leftarrow 3Dsensor_Capture()\)
6. Transform \(T \leftarrow \text{Compute_Transformation} (I(t), I(t-1))\)
7. if \(T\) is NULL:
8. Display “Could Not Compute Motion” 
9. else:
10. \(\tau(t+1) \leftarrow \tau(t) \times T\)
11. \(x_{t+1}, y_{t+1}, \theta_{t+1} \leftarrow \text{Generate_3Df(Pose)}\)
12. Display \(x, y, \theta\)
13. \(t \leftarrow t+1\)

The function Compute_Transformation takes as input the previous image and the live image and returns the camera pose transformation matrix. \(\tau\) is the trajectory of the camera, stored as a set of discrete poses. \(I(t)\) is the input image at time \(t\). The initial position is \((x_0, y_0)\) and the initial orientation is \(\theta_0\). The performance of visual odometry is good for short distance, however, in the case of longer runs the cumulative error becomes too large and we need to reinitialize the location of the robot. We suggest reinitializing using the output of experience based localization.
3.3 Histogram of oriented gradients

Histogram of oriented gradients (HOG) is a feature descriptor explained by Dalal and Triggs [33]. This feature with SVM is used for visual place recognition. HOG is calculated by first calculating gradient images and making histograms using direction range as bin and magnitude as frequency. Figure 6 shows the process.

4 Methodology

In order to carry out localization a map needs to be known, which for this problem corresponds to a topological map of the environment. Difficulty or error arises when the map is not accurate. Given an input image we predict the starting point using visual place recognition. Two methodologies are used. In the first, we separately applied visual odometry and experience based localization; and compared the two methodologies with each other. In the second methodology, we applied a fusion of experience based localization and visual odometry. Figure 7 shows a Kinect sensor attached to a Firebird 6 robot.

4.1 Visual place recognition

The problem of visual place recognition is to find out which ‘place’ or geographical location the robot is in by looking at the surroundings. For localization, we first need to find where the robot currently is and what its starting position is. Consider the situation where a human wakes up and the first thing he/she does is to observe his/her surroundings to localize himself/herself. We use the same approach. Finding the starting position is a challenging task as the starting position result affects the results of visual odometry.

Let a robotics area have \( n \) places called \( \rho_1, \rho_2, \rho_3 \ldots \rho_n \) while the robot only has a monocular camera that takes an image \((I)\). Visual place recognition is the classifier \( C_I(I) = \rho_i \) that takes an image as input and returns the place label.
as an output. Because the image may be very large in size, we have used HOG descriptors as a feature to train a SVM classifier \([34]\), making the classifier \(C_2(\text{HOG}(I))=p_i\).

Knowing the place, one needs to know the starting position inside the place or room. The problem is to find \((x_0, y_0, \theta_0)(x_0, y_0)\rho_i\), given the sensory percepts of the robot. A typical way to do so would be to make a database \(D(\rho_i)=\{(x, y, \theta)\}\) where the image \(I\) is the input and \((x, y, \theta)\) is the labelled output. A classifier \(C_3(I)=(x_0, y_0, \theta_0)(x_0, y_0)\rho_i\) can then be made to solve the problem of start point computation. The dataset \(D\) may be made recording all possible \((x, y, \theta)\) combinations. However the assumption here is operation in a wide open hall or room wherein the robot motion is not restricted on any axis, unlike roads wherein there is a very small lateral dispersion. In such contexts sometimes by a single image we may not get adequate features to compare and that may result in poor results. Say the image to be compared is very near to that of a plain wall that has no features. Sometimes even we cannot identify the correct location by just looking at a wall. So we just turn or rotate a little in order to correctly identify the starting location. Using the same motivation we take multiple images at an angular dispersion from the same \((x_0, y_0)\) location. Taking multiple images from the same coordinate can get sufficient features and increase accuracy.

To find the starting point we take 6 images at a difference of \(\pi/3\) radians each. Each of the images is searched in the desired room database recorded earlier and 1 Nearest Neighbor classifier is used to find the resulting image. Image matching is done by finding the ORB (Oriented Fast and Rotated Brief) descriptors \([35]\) and using FLANN \([36]\) library to find approximate nearest neighbors. Let \(C_4(\text{ORB}(I))=(x_j, y_j, \theta_j)(x_j, y_j)\rho_j\), where \(I\) is the image number taken in intervals of \(\pi/3\) radians. All classifications \((x_j, y_j, \theta_j)\) corresponding to each of the image is recorded. The final starting position is taken as the one that is the most centrally placed, or the one that minimizes the metric given by (3). In (3) \(d()\) is the distance function. In short, we find the most centrally located point among the six points.

\[
(x_0, y_0, \theta_0) = (x_j, y_j, \theta_j),

j = \arg \min_k \sum_{l:k} d([x_k, y_k]^T, [x_l, y_l]^T)
\tag{3}
\]

The choice of the classifier is again due to the nature of the problem. In order to create a dataset we visited all the places once and recorded various poses at every coordinates creating a single experience of visiting every possible orientation of the robot. Unlike the literature, the map is wide open and creation of a dataset requires a huge effort due to multitude of positions possible for the robot. Every image is mapped to a single coordinate and hence the classifier.

### 4.2 Experience based localization

For experience based localization, the map first needs to be converted into a topological map. The underlying map represents wide indoor space, which does not naturally represent a topological map. However to embed a graph into the same, all possible locations with some uniform sampling are taken as the vertices, \(V = \{(x, y, \theta)\}\). Two vertices are connected to each other by an edge if they are neighbors of each other using a neighborhood function \(\delta\). \((v_i, \delta(v_j))E\). It was easy to create a topological map. The only conditions we need to keep in mind was that the robot/machine can go anywhere except on reaching the limits in X and Y directions. There were points which were inaccessible so complete black images were mapped to these points meaning it is a wall or obstacle.

The initial position \((x_0, y_0, \theta_0)\) is known from the visual place recognition algorithm. Assume that the position at any time \(t\) is known to be \((x_t, y_t, \theta_t)\) which is covered by the vertex \(v_t\) of the sampled topological graph. The robot should be able to localize itself as it moves to the next position or as it takes a step. As per modelling, the maximum that the robot can move within a time iteration is less than the vertices being covered by the neighborhood function, that is \(d([x_t, y_t, \theta_t]^T, [x_{t+1}, y_{t+1}, \theta_{t+1}]^T) < \max_{v_i\delta(v_j)}(d(v_i, v_j))\). Hence, knowing the position at time \(t\) to be in the vertex \(v_t\), the position at time \(t+1\) may be at either of the vertices in \(\delta(v_t)\). We only need to check for this range of coordinates. A topological map for an obstacle-free region is shown in Figure 8. For simplicity Figure 8 does not show the orientation axis. Figure 9 shows the coordinate axis.

The choice of the neighborhood function \(\delta(v_t)\) is such that any two vertices at a geometric distance of \(2\sqrt{2}\) or less will be connected, irrespective of the orientation, that is \(d(v_t, x_{t+1}, y_{t+1}, \theta_{t+1}) = d([x_t, y_t, \theta_t]^T, [x_{t+1}, y_{t+1}, \theta_{t+1}]^T) < 2\sqrt{2}\). Here \(d\) is the Euclidian distance function. From Figure 8 we see that there are 25 coordinates in the XY place. Each coordinate has 6 images mapped to it in the orientation plane, so there are 150 images that needs to be matched with the current online image. The number will be larger if there are more than one image per location. A localizer takes the online image and compares it with these 150 images from experience to give the best possible current location. Let \(D(x, y, \theta)\) denote the image corresponding to an output entry \((x, y, \theta)\) in the database. Since \((x, y, \theta)\) is dis-
Experience based localization in wide open indoor environments

Figure 8: Grid Map. Suppose the black dot is a previously visited coordinate then its neighbors are the green dots. 1 unit in X and Y is equal to ‘a’ which is the distance between recorded images.

Figure 9: Coordinate axis with robot orientations. All $\theta$ values around a single coordinate are displayed. $\theta$ is the rotation around the Z-axis. The Z-axis is normal to X-Y plane.

cretized in the dataset, the function represents a hash table with $(x, y, \theta)$ as the key and the image as the value. Algorithm 2 shows how the algorithm is carried through a pseudo code. In the algorithm the term position and vertex are used interchangeably even though the former is a continuous quantity and the latter is a discrete quantity because all operations returning position are only capable of returning discrete outputs corresponding to which a vertex always exists. Figure 10 summarizes the algorithm of Experience based localization with the use of a flowchart.

Algorithm 2: Localization
1. $\tau(0) \leftarrow (x_0, y_0, \theta_0), t \leftarrow 0, v_0 \leftarrow (x_0, y_0, \theta_0)$
2. while (3Dsensor is ‘ON’):
3. $I(t) \leftarrow 3DSensor\_Capture()$
4. Compute $\delta(v_t)$
5. $v_{t+1} \leftarrow v_j, j = \arg \min_{v_i} \delta(v_i)(d(ORB(I(t)), ORB(D(v_j))))$
6. $\tau(t+1) \leftarrow v_{t+1}, t \leftarrow t+1$

Figure 10: Flowchart of experience based localization. This figure shows the process of experience based localization for robot pose estimation.

4.3 Combined approach

The previous sections presented the use of experience based localization and visual odometry independently. In this section we find the best matched coordinate of the online image using experience based localization and then we use the location with the online image to get better results. Visual odometry is a very good mechanism for localization, however the errors keep getting accumulated and that may result in a very poor localization over the long run. However, the output is continuous in nature. The experience based localization finds the location from a dataset using live images and hence the errors do not easily accumulate. However the output is limited to the stored experiences and in this case discrete in nature. By fusing the two techniques we get rid of the additive nature of the errors of localization using experience based localization for macro localization and get a continuous output based on the visual odometry for micro localization. Furthermore, the database of experience based localization can also aid in visual odometry by providing better images to be used to compute the transformation. The betterment of the approach is illustrated in Figure 11. If the original location of the robot is indicated by the red symbol, the current location by the green symbol and the resulting location obtained by using experience based localization by the blue symbol, then computing visual odometry between the green and blue poses gives a better re-
result compared to computing visual odometry between the green and red poses. First we detect the robot’s position using experience based localization and after that, in order to find out the live location of the robot, we compute the pose of the robot from applying visual odometry between the live image and the image obtained from experience based localization.

Figure 11: The different orientations of the robot. The arrowhead indicates the direction of the robot.

Figure 12 summarizes the algorithm discussed through a flowchart. The hybrid algorithm is given by Algorithm 3. Visual odometry takes two images and computes the distance between them. Algorithm 3 is a slight modification of Algorithm 2.

Algorithm 3: Combined approach
1. \( \tau(0) \leftarrow (x_0, y_0, \theta_0), t \leftarrow 0, v_0 \leftarrow (x_0, y_0, \theta_0) \)
2. while (3DSensor is ‘ON’):
3. \( I(t) \leftarrow 3DSensor\_Capture() \)
4. Compute \( \delta(v_t) \)
5. \( v_{t+1} \leftarrow v_j = \arg \min_{j} \delta(v_j)(d(ORB(I(t)), ORB(D(v_j)))) \)
6. \( \tau' \leftarrow v_{t+1} \)
7. \( \Delta \leftarrow Visual\_Odometry\_Distance(I(t), D(v_{t+1})) \)
8. \( \tau(t+1) \leftarrow \Delta \times \tau' \)
9. \( t \leftarrow t+1 \)

The function \( D(v_{t+1}) \) returns the image corresponding to the location given by experience based localization. The image is fed as input to the function Visual\_Odometry\_Distance which returns the transformation between the live image and image obtained by experience based localization.

5 Dataset
Since no standard dataset was suited for our approach, the data set had to be exclusively created by us. The problem being solved was specific in nature which was to operate a robot in the robotics arena of the institute. Hence only a dataset of the arena was taken. As per institute policies only the arena could be physically hand-annotated (causing physical changes to the area) and the robots could be operated only in the same region. That said, the robotics arena of the institute is itself very large and diverse. The area in question was exhaustively covered by considering all positions and orientations. We did everything possible in order to record a rich dataset out of the resources available in the need of the broader project. The dataset consists of \(640 \times 480\) pixel images recorded by a Kinect sensor at a height of about 107 cm above the floor. The dataset has been recorded in three different rooms: Classroom, Lab and Office. The floor of each of the rooms is a 2-D surface (XY) that does not have any variations in the Z-axis (normal to the floor). At each coordinate \((x, y)\) six images at an angle \(\pi/3\) from 0 to \(2\pi\) clockwise. The classroom has \(18 \times 20\) coordinates consisting of a total of 2160 monocular images. The lab has \(16 \times 30\) coordinates consisting of total 2880 monocular images. The office has \(8 \times 10\) coordinates consisting of total 400 monocular images (in the office dataset we recorded on intervals of 72 degrees). We also have also depth images recorded in the same manner as above. The lab dataset was not recorded on a single night. Frequent changes in the positions of books, bottles and some other minor changes occurred during the recording.
of the dataset. Figure 13 shows an example of the recorded dataset for the coordinate (11, 5) in the lab dataset.

![Dataset Images](image)

Figure 13: Dataset Images: (a), (b), (c), (d), (e), (f) shows images at 0 degree, 60 degree, 120 degree, 180 degree, 240 degree and 300 degrees respectively at the coordinate (11, 5) from the lab dataset.

6 Experimental results

All three datasets have some minor changes. We have used the root mean square (RMS) as our evaluation metric as shown in (4).

\[ \text{rms} = \sqrt{\frac{\sum_{i=1}^{N} \text{error}_i^2}{N}} \]  \hspace{1cm} (4)

Another metric used for evaluation is the misclassification accuracy, wherein a result is termed as correct if the error is less than the resolution of recording. The development was done in C++ and Python. We used a system with Ubuntu 16.04 and 8 GB RAM.

6.1 Results of experience based localization

Table 1 shows the results of the experience based localization. Let \( a \) is the length of the coordinate axis or the resolution by which the dataset was recorded. No two images exist at a distance smaller than \( a \). For the classroom and lab \( a \) is 38 cm and for the office \( a \) is 50 cm. All results are reported relative to this distance to make the results invariant of the recording resolution. hit(1a) denotes the probability of correctly classifying grid within \( a \) distance, that is the resolution of recording the dataset. For the office dataset out of 33 points 8 points are correctly classified within \( a \) distance. hit(2a) denotes the probability of correctly classifying grid within \( 2a \) distance. For office dataset out of 33 points 26 points are correctly classified within \( 2a \) distance.

We see that the RMS error of the lab dataset is greater than the other datasets. This is due to some minor changes such as positions of mouse, books and keyboards changed while recording the dataset. Additionally, some of the computer screens were ‘on’ in initial recording of dataset and ‘off’ during the later recordings of the dataset. We tried to keep the position of chairs the same during the whole recording of the dataset.

6.2 Results of visual odometry

Table 2 shows the results of the visual odometry experiment. The first row shows the overall performance of the test runs while the second row shows the performance of first ten steps taken by the robot. We see that the RMS is relatively low for the first ten points and keeps increasing thereafter due to accumulation of errors. Thus for a long run we need to reinitialize the visual odometry location with the pose obtained from experience based localization so as to restrict the cumulative error from becoming too large.

6.3 Results with dynamic obstacles

We tested experience based localization and visual odometry algorithms for dynamic obstacles and the results were 4.23 textita and 2.92 \( a \) respectively. The visual odometry results show twice as much as error in normal case. The reason is that it may not get more similar features in consecutive frames because of the dynamic obstacles. This can lead to worse results than the normal case. But 2.92 \( a \) is approximately 1.1 m which means it is still able to localize itself over a rough estimation.

6.4 Results of initial point

The RMS error of initial position by visual place recognition is 0.81 \( a \). It is small because we are taking an image
Table 1: The RMS of experience based localization experiment.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>RMS(a)</th>
<th>hit(1a)</th>
<th>hit(2a)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Office</td>
<td>1.31152</td>
<td>8/33</td>
<td>26/33</td>
</tr>
<tr>
<td>Classroom</td>
<td>1.34602</td>
<td>14/33</td>
<td>28/33</td>
</tr>
<tr>
<td>Lab</td>
<td>1.39014</td>
<td>15/20</td>
<td>17/20</td>
</tr>
</tbody>
</table>

Table 2: The results of visual odometry experiment.

<table>
<thead>
<tr>
<th>Steps</th>
<th>RMS (m)</th>
<th>RMS (a)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall</td>
<td>0.5407</td>
<td>1.421</td>
</tr>
<tr>
<td>first 10</td>
<td>0.3235</td>
<td>0.842</td>
</tr>
<tr>
<td>11 to 20</td>
<td>0.6051</td>
<td>1.578</td>
</tr>
<tr>
<td>after 20</td>
<td>0.6213</td>
<td>1.631</td>
</tr>
</tbody>
</table>

at every \(\pi/3\) radians difference rotated clockwise and then finding the most centrally located point among them. It has to be small because it is used to initialize the odometry module and its accuracy is very important in ensuring all the following algorithms works accurately.

### 6.5 Results of visual place recognition

There are 300 training images in the dataset. There are 93 testing images. All testing images are different from training images. The accuracy is calculated using precision \(p\) and recall \(r\). Accuracy is given by (5), precision is given by (6) and recall is given by (7). Table 3 summarizes the results for all three classes (Classroom, Lab and Office).

\[
\text{accuracy} = \frac{2pr}{p+r} \quad (5)
\]

\[
p = \frac{\text{truepositives}}{\text{truepositives} + \text{falsepositives}} \quad (6)
\]

\[
r = \frac{\text{truepositives}}{\text{truepositives} + \text{falsenegatives}} \quad (7)
\]

### 6.6 Results of the combined approach

The RMS error of using experience based localization with visual odometry is 2.32 \(a\). The error is a little large because visual odometry requires good initialization but the problem is that we are not able to provide good initial orientation by localization. The poor resolution of recording the dataset in terms of orientation is the limiting factor. Nevertheless, as apparent otherwise from the results, the combined approach is able to give realistic location estimates by using the combination of both methodologies. There was no large growth in error as was dominant in visual odometry. The system could interpolate between points and was not restricted to giving outputs between the points where experience was recorded. It is possible to record an experience dataset with a very high resolution, but it would take a significant amount of time. In a more realistic setting, the experience dataset will be a lot coarser and the benefits of the combined approach will be much larger.

### 7 Conclusion

For localization, visual odometry works well for small distances that are suitable for indoor localization. However, it needs to be initialized accurately in a prebuilt map and errors may accumulate with time. This paper used visual place recognition for setting an initial point. A naïve implementation of visual place recognition gives poor performance for wide-open scenarios as the results are based on a single image. This paper proposes the use of multiple images at different orientations for improved performance. Experience based localization performs well in static maps with non-cumulative errors, while visual odometry performs well in dynamic maps and gives continuous outputs to localization. This paper proposed the fusion of these two methodologies. The resultant algorithm performed well, while the results could have been even better if more accurate orientation was predicted from experience based localization using a better resolution of orientation while recording the dataset.
The biggest limitation of this project is that we are recording the dataset manually. In the future we can design a robot which when left in a room can make the entire dataset. The current algorithm is also highly susceptible to changes in the illumination of the room and for large changes in the environment we have to recreate the database. We can in the future modify the algorithm such that changes in the environment are incrementally updated in the original database. If we take the camera at a low height the results for both visual odometry and experience based localization are very poor as a lot of ground features are detected which are difficult to match. The dataset cannot be shared across robots as different robots have different heights. It should be possible to make adaptations to changing robots. All of these limitations can be taken care of or the effect be mitigated by making changes to the algorithm in the future.

### References


[8] C. Harris, M. Stephens, A combined corner and edge detector, Plüss Research Roke Manor, United Kingdom, 1988


[23] S. Li, P. Kar, Context-aware bandits, CoRR abs/1510.03164, 2017


[27] S. Park, K. S. Roh, Coarse-to-fine localization for a mobile robot based on place learning with a 2-D range scan, IEEE Transactions on Robotics, 2016, 32(3), 528-544


[31] L. Quan, Z. D. Lan, Linear n-point camera pose determination, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1999, 21(8), 774-780


